

REMARKS

This application has been carefully considered in connection with the interview granted on October 17, 2006. Reconsideration and allowance are respectfully requested in view of the following.

Summary of Rejections

Claims 1-27 were pending at the time of the Office Action.

Claims 1-27 were rejected under 35 U.S.C. 102(b) as being anticipated by Gwak et al.

Summary of Response

Claims 1-19 and 21-27 were canceled.

Claim 20 was amended.

Claims 28-46 were added.

Remarks and Arguments are provided below.

Summary of Claims Pending

Claims 20 and 28-46 are currently pending following this response.

Examiner Interview on October 17, 2006

Applicant thanks Examiner Mofiz for granting the in-person interview on October 17, 2006. In the interview, it was suggested to include limitations from claims 20, 35, and 40 in each independent claim. In accordance with the suggestion, independent claim 20 has been amended to further include limitations that were presented in new claims 35 and 40 of the response filed

on October 4, 2006. In particular, claim 20 has been amended from the response filed on October 4, 2006 as indicated in bold.

20. (Currently Amended) A method of generating test data to test an application, the method comprising:

~~identifying a data field and a data value for the data field to test an application;~~

~~defining in a property file a set of data field identifiers with a data value corresponding to each of the data field identifiers;~~

~~providing a data definition including the data field and the data value to a property file;~~

~~defining in the property file at least one test string format as a sequence of a subset of the data field identifiers in the property file,~~

wherein the definitions in the property file do not adhere to a COBOL copybook syntax;

~~providing a [[n]] command prompt instruction for **managing generating** a test file;~~

~~generating, based on an instruction, a string of test data the data field and data value to the test file using based on the instruction, wherein the string of test data is **structurally equivalent to a data**~~

string output by a COBOL application and includes a sequence of the data values that are not delimited by any special character and arranged in accordance with the sequence of corresponding field identifiers defined by one of the test string formats in the property file, ~~the test file used to test the application;~~

~~testing the application with the test file; **and**~~

modifying the contents of the test file with a change test file command prompt instruction.

Claims 30 and 35 presented in the response filed on October 4, 2006 have not been included in this supplemental response. As a result, claims 31-34 have been renumbered herein

as claims 30-33 respectively, and claims 36-48 have been renumbered herein as claims 34-46 respectively.

Claim 40 has been renumbered as claim 38 as noted above and amended from the response filed on October 4, 2006 as indicated in bold.

38. (New) A method of generating test data to test an application, the method comprising:

defining in a property file a plurality of data field identifiers with a data value corresponding to at least one of the data field identifiers, and at least one test string format as a sequence of the data field identifiers, **wherein the definitions in the property file do not adhere to a COBOL copybook syntax;**

generating a first test file corresponding to the test string format based on a new test file command prompt instruction, wherein the first test file is created to include a string of the data values **that is structurally equivalent to a data string output by a COBOL application and the data values** are not delimited by any special character **and are arranged** in accordance with the sequence of corresponding field identifiers in one of the test string formats;

testing the application with the test file; and

modifying the contents of the first test file with a change test file command prompt instruction.

As noted in the response filed on October 4, 2006, Gwak discloses generating test data that is time-consuming and complex using an automatically test data generator. The automatic test data generator of Gwak uses structure information, content information, and raw data to generate test data that conforms to a particular format, namely MPEG-2. Applicant recognizes the similar motivations for desiring to automatically generate test data between the instant disclosure and the Gwak reference. However, in the instant disclosure there are many differences

with the Gwak reference in the format of the test data that is output, how data that is used by the test data generator is structured, and the functions of the test data generator, such as being able to easily edit existing test data. These differences are described in more detail below.

Format of Output Test Data

The present disclosure is particularly directed to generating and editing test data that conforms to the format of the output of COBOL copybook definitions. The output of COBOL copybook definitions may include a very long data string of characters where different data fields in the string are not delimited by any special characters. Since the data in the string is not delimited, it is very hard for a person to read and keep track of where they are in the string. This may result in errors in generating and editing a string due to losing your place or interpreting a sequence of characters as a data field that actually spans different data fields. The lack of readability of these types of data strings may be particularly appreciated by looking at the short data string in the last line of paragraph 0025 of the instant disclosure. In paragraph 0017 of the instant disclosure an example of editing such a data string is described wherein the manual process described is difficult, time consuming, and subject to error. Independent claim 20 and new independent claim 38 include this difference with the limitation that the test data generated is a string of “data values that are not delimited by any special character”.

Structure of Data Used by Test Data Generator

Since the output test data is a string of data values that are not easily readable, it is desirable to structure the definition of the format of the output test data in a readable manner. This enables an easier creation and editing of a test data format definition. Paragraph 0023 of the instant disclosure describes an exemplary property file that may be used by the test data generator

to generate the string of test data. At the top of the exemplary property file is a plurality of data field identifiers that are COBOL data definitions. In this example, each of the data field identifiers are assigned a data value. For example, the PON COBOL data definition is assigned the value PON598. Independent claim 20 has been amended to include this difference with the limitation of “data field identifiers with a data value corresponding to each”. It is noted that not every data field identifier has to be assigned a data value. For example, as disclosed in paragraph 0036 of the instant disclosure, data field identifiers without a data value may be assigned a null value. New independent claim 38 includes this difference with the limitation of “data field identifiers with a data value corresponding to at least one of the data field identifiers”.

In the middle of the exemplary property file is a plurality of output data string format definitions. Each data string definition is named and is defined as a sequence of data field identifiers. Defining the data string using the data field identifiers enables one skilled in the art to more easily read and understand what is contained in the data string. Due to the use of COBOL data definitions that are recognizable and readily identifiable by one skilled in the art, the format definition may be more easily created and edited. Independent claim 20 and new independent claim 38 include this difference with the limitation that the “test string format” is defined “as a sequence of ... the data field identifiers”.

At the bottom of the exemplary property file is an indication of which test strings are to be output. As disclosed in paragraphs 0025 and 0030 of the instant disclosure, a separate file is generated for each test string that is indicated to be output. Independent claim 20 and new independent claim 38 include this difference with the limitation that a (singular) test file is generated for “one of the test string formats”. The structure of the property file as described

above is particularly well suited and useful for generating and editing long strings of characters that are not delimited by any special characters. Also, the structure of the property file as described above is particularly well suited for editing the test data format definition as described in paragraph 0031 of the instant disclosure.

Functions of Test Data Generator

The test data generator of the present disclosure also includes a number of useful functions that enable the test data to be easily generated and edited. As disclosed in paragraph 0036 of the instant disclosure, command line instructions may be input to either generate a new test data file or change an existing test data file. When changing an existing test file, a command line argument may be included with the change command to assign a new value to any of the data field identifiers instead of using the value in the property file. Only the values in the test file corresponding the newly defined data field identifiers in the command line argument are changed. This provides a fast, easy, and accurate way of editing the long sequences of characters produced by the test data generator of the instant disclosure. As disclosed in paragraphs 0028 and 0029, when editing a large number of data field identifiers, an empty properties file may be generated wherein only those data field identifiers that are to be changed may be edited to have a desired new value. Independent claim 20 and new independent claim 38 includes this difference of editing an existing test file with the limitation of “modifying the contents of the first test file with a change test file command prompt instruction”.

COBOL Data Generator

As disclosed in paragraph 0033, the properties file of the instant disclosure may be used by the test data generator to produce a string of test data that is structurally equivalent to a data

string output by a COBOL application, but the definition of the test string format in the properties file does not adhere to a COBOL copybook syntax. Independent claim 20 and new independent claim 38 have been amended herein to also include this difference.

Conclusion

Applicant respectfully submits that the present application is in condition for allowance for the reasons stated above. If the Examiner has any questions or comments or otherwise feels it would be helpful in expediting the application, he is encouraged to telephone the undersigned at (972) 731-2288.

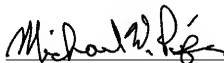
The Commissioner is hereby authorized to charge payment of any further fees associated with any of the foregoing papers submitted herewith, or to credit any overpayment thereof, to Deposit Account No. 21-0765, Sprint.

Respectfully submitted,

Date: _____

10/23/2006

CONLEY ROSE, P.C.
5700 Granite Parkway, Suite 330
Plano, Texas 75024
(972) 731-2288
(972) 731-2289 (facsimile)



Michael W. Piper
Reg. No. 39,800

ATTORNEY FOR APPLICANT